REMARKS

Applicant is in receipt of the Office Action mailed December 11, 2007. Claims 1-29 are pending in the case. Reconsideration of the present case is earnestly requested in light of the following remarks.

Telephone Interview Summary

On Wednesday, January 30, 2008, a telephone interview was conducted with the Examiner by Mark S. Williams (Reg. No. 50,658). The prior art references of Xavier and Takagi were discussed, as well as the technical distinctions between data flow diagrams and scene graphs. The Examiner suggested amending the independent claims to clarify that the generated scene graph is different and distinct from the nodes of the data flow diagram that generates the scene graph, and indicated that this would overcome the cited art. Applicant further suggested amending the independent claims to include the feature of storing the scene graph after its generation. Applicant agreed to send a proposed set of amended claims to the Examiner for consideration. It is Applicant's understanding that these amendments bring the case into condition for allowance.

In a subsequent telephone conversation held on Wednesday, February 27, the Examiner suggested that Applicant submit an Office Action Response with the amendments discussed. Applicant agreed to do this.

Section 103 Rejections

Claims 1-3, 10, and 21-29 were rejected under 35 U.S.C. 103(a) as being unpatentable over Xavier et al (US Pat. Pub. 2003/0079207, "Xavier") in view of Takagi et al (US Patent No. 6,493,001, "Takagi"). Applicant respectfully traverses the rejection.

Amended claim 1 recites:

1. A computer-accessible memory medium comprising program instructions for creating a scene graph, wherein the program instructions are executable to implement:

creating a data flow diagram in response to input, wherein said creating comprises:

displaying a first plurality of nodes on a display, wherein each of the plurality of nodes is executable to create at least a portion of the scene graph;

connecting the first plurality of nodes to create the data flow diagram, wherein the first plurality of nodes are connected to specify data flow among the plurality of nodes;

executing the data flow diagram, wherein said executing creates the scene graph; and

storing the scene graph in a memory medium after said executing;

wherein the scene graph comprises nodes representing corresponding objects in a scene, wherein the nodes representing the objects are different from the nodes in the data flow diagram, wherein the scene graph specifies a plurality of objects and relationships between the objects, and wherein the scene graph is usable in rendering a graphical image of the plurality of objects.

Nowhere does the cited art disclose **creating a data flow diagram in response to input, wherein said creating comprises: displaying a first plurality of nodes on a display, wherein each of the plurality of nodes is executable to create at least a portion of the scene graph**, as recited in claim 1.

As explained in the previous Response, cited Figures 1, 4, 5, and paragraphs 24, 27, and 57, disclose data-flow based simulation, but fail to disclose *nodes that are executable to create at least a portion of a scene graph.*

For example, Figure 1 is a diagram illustrating a conventional data-flow-based modular simulation of an exemplary robot arm.

Figure 4 illustrates a data-flow-based modular simulation of an exemplary system including sensors and a sensor world model. Each module in the system is considered a proxy for, i.e., represents or simulates, a corresponding entity.

Figure 5 illustrates an exemplary meta-module. More specifically, Figure 5 illustrates "a representation of an individual mobile robot modeled as an Umbra embodied agent meta-module", which, again, is a simulation module for simulating a mobile robot.

Paragraph [0024] is directed to simulations of systems or phenomena that include a large and/or undetermined number of devices or components, where each simulation module includes multiple I/O for connecting to each of the other modules in the simulation.

Paragraph [0027] describes a more sophisticated approach to interconnecting simulation modules, specifically, using a switching module to configure I/O among modules, thereby reducing a quadratic number of connections to a linear number.

Paragraph [0057] discloses communication modules for the simulation that model various communication mechanisms and dependencies among the devices or components being simulated.

As may be seen, none of the cited portions of Xavier teach or suggest nodes that are executable to create or generate (at least a portion of) a scene graph. In other words, none of the modules described in Xavier are described as being executable to create or generate a portion of a scene graph. In fact, the only mention Xavier makes of a scene graph at all is in paragraph [0035], which reads as follows:

> [0035] In exemplary Umbra embodiments multiple worlds are common. Examples of Umbra worlds include a radio communications world, various sensor/sensing worlds, terrain-based mobility dynamics world, and a multi-body contact dynamics world. In addition, Umbra scene-graph used by an Umbra visualizer may typically be integrated into Umbra simulations as part of a world module.

Applicant respectfully notes that according to Xavier, a scene-graph may be integrated into the simulations, e.g., as part of a world module, but Xavier is silent regarding generation of the scene graph, and never describes or even hints at nodes or modules in a data flow diagram that are executable to create or generate a scene graph.

Thus, Xavier fails to teach or suggest these features of claim 1.

Nor does the cited art disclose **executing the data flow diagram, wherein said executing creates the scene graph**, as recited in claim 1.

The Office Action asserts that Xavier discloses this feature in paragraphs 37, 40, and 50, which read:

[0037] As will be appreciated by those skilled in the art, data movement between modules in conventional data-flow-based simulation systems is through connections between modules in the data-flow graph. In conventional systems, if there are n entities that can be directly affected by or can directly cause effects in a given interaction phenomenon, then generally either: a) the meta-module for each such entity must have data flow connections to meta-modules of the other such entities, resulting in o(n.sup.2) data-flow connections in the simulation data-flow network; or b) the interaction phenomenon may be separately simulated by a module, e.g., a "switch" as described above, with o(n) input ports and output ports connected to modules in the n meta-modules simulating the n entities. Other characteristics of conventional methods limit the flexibility of simulations particularly with regard to interaction with a varying number of devices and/or interfaces.

[0040] Further, it is not required that a world module be a part of the data-flow graph associated with a simulation, however update order of the non-world modules in the simulation is important. Thus for convenience, world modules are in most circumstances made a part of the data flow graph. To an exemplary Umbra simulation kernel in accordance with the present invention, world modules are treated as other modules.

[0050] Exemplary Umbra embodiments can be integrated with collision detection libraries as described in University of North Carolina (UNC) at Chapel Hill's V-Collide system as further described in V-COLLIDE: Accelerated Collision Detection for VRML, Hudson, T. C.; M. C. Lin, J. Cohen, S. Gottschalk, and D. Manocha, Proceedings VRML '97, Monterey, Calif., February 1997, and also as in Sandia National Laboratories' C-Space Toolkit as described in Fast Swept-Volume Distance for Robust Collision Detection, Xavier, Patrick, IEEE International Conference On Robotics and Automation, Albuquerque, N.Mex., Apr. 20-25, 1997. Unlike the SMART integration, the V-Collide and C-Space tools can be integrated at the "world" level. For example, in accordance with integrated embodiments of the present invention, geometric computation worlds use a database of scene and other geometry to compute intersections between geometric objects. Accordingly, V-Collide may be used for its high speed analysis in static environments. For example, non-contact sensor simulation modules compute whether cones that represent the sensed region intersect with the surrounding environment. The C-Space Toolkit uses high speed analysis of swept volume intersections. An example would include rapid motion planning for articulated robot manipulators.

As may be seen, and as discussed in the previous Response, which is hereby incorporated by reference, these paragraphs make no mention of a scene graph at all. Paragraph 50 does mention a scene, but only in the context of a scene (and other geometry) database used by geometric computation worlds. None of these citations (nor Xavier in general) discloses executing a data flow diagram to create a scene graph, as claimed.

The Office Action states that "Xavier et al do not go into the details that the executing of the graphical program creates the scene graph per se, but do show efficiently accessing the program to render the scene graph." Applicant respectfully submits that Xavier nowhere discloses executing a data flow diagram to create a scene graph, as mentioned above, and that Xavier's "efficiently accessing the program to render the scene graph" is not germane to the patentability of claim 1, at least for the reason that rendering a scene graph (actually, rendering a scene using a scene graph) is not equivalent to creating a scene graph.

The Office Action then asserts that Takagi shows "executing of the graphical program to create the scene graph", citing Figures 4, 9, 10, col.10:20-50, col.11:29-62, and col.12:1-25.

Applicant has reviewed these citations carefully, and can find no illustration or description of this feature.

For example, cited col.10:20-50 is directed to using a browser emulator to verify scene graph functionality for contents of a scene graph database, but makes no mention of executing a data flow diagram to create a scene graph.

Cited Figure 4, col.11:29-62, and col.12:1-25 are directed to a menu screen displaying a contents creation tool, and disclose a scene-graph window operable to display the nodes (of a scene graph) in a current scope, which means those nodes referenced to be edited, e.g., using a scene-graph edit tool, as well as a world information window for specifying other attributes of a scene, e.g., background, world information, navigation information, and a conductor window for managing and verifying scene content and operations. Nowhere do these citations discuss or suggest executing a graphical data flow diagram to create a scene graph. Rather, as Takagi makes clear, the

user edits the scene graph nodes manually using the scene-graph edit tool. Applicant notes that the tree diagram of scene graph nodes described in Takagi is not a data flow diagram executable to generate a scene graph (e.g., scene graph nodes), but rather, is a display of scene graph nodes themselves, and is used to manually edit the nodes.

Cited Figures 9 and 10 are directed to adding an avatar (from a resource library) to an avatar page of the multi-user window as an avatar that can be used by the user in the multi-user world being edited.

As may be seen, none of these citations discloses this claimed feature. Nor do Xavier and Takagi in general teach or suggest this feature.

Moreover, nowhere do Xavier and Takagi disclose **storing the scene graph in a memory medium after said executing; wherein the scene graph comprises nodes representing corresponding objects in a scene, wherein the nodes representing the objects are different from the nodes in the data flow diagram, wherein the scene graph specifies a plurality of objects and relationships between the objects, and wherein the scene graph is usable in rendering a graphical image of the plurality of objects**, as recited in claim 1.

In other words, the cited art fails to teach or suggest storing a scene graph that has been generated via execution of a data flow diagram, where the scene graph comprises nodes representing corresponding objects in a scene, wherein the nodes representing the objects are different from the nodes in the data flow diagram, i.e., where the nodes of the scene graph are separate and distinct from the graphical program nodes of the data flow diagram.

Applicant respectfully notes that in Xavier, the data flow simulation diagram may include a "world module" that defines world attributes which a simulation may operate under, and that a (n Umbra) scene-graph may be included in a world module for use by a (n Umbra) visualizer to render the scene or world. In Takagi, tools are provided for displaying nodes of a scene graph in a tree diagram, and manually editing them, but Takagi does not mention generating a scene graph via execution of a data flow diagram. More specifically, nowhere do Xavier and/or Takagi describe or even hint at a graphical data flow diagram with graphical program nodes that are executable to generate a scene

graph, *where the scene graph comprises nodes representing corresponding objects in a scene, and where the nodes representing the objects are different from the nodes in the data flow diagram*.

Thus, the cited art fails to teach or suggest these features of claim 1.

Thus, for at least the reasons provided above, Applicant submits that Xavier and Takagi, taken singly or in combination, fail to disclose all the features and limitations of claim 1, and so claim 1 and those claims respectively dependent therefrom, are patentably distinct and non-obvious over the cited art, and are thus allowable.

Claims 25, 28, and 29 include similar limitations as claim 1, and so the above arguments apply with equal force to these claims. Thus, for at least the reasons provided above, Applicant submits that claims 25, 28, and 29, and those claims respectively dependent therefrom, are patentably distinct and non-obvious over the cited art, and are thus allowable.

Removal of the section 103 rejection of claims 1-3, 10, and 21-29 is earnestly requested.

Applicant also asserts that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

Applicant submits the application is in condition for allowance, and an early notice to that effect is requested.

If any extensions of time (under 37 C.F.R. § 1.136) are necessary to prevent the above-referenced application(s) from becoming abandoned, Applicant(s) hereby petition for such extensions. The Commissioner is hereby authorized to charge any fees which may be required or credit any overpayment to Meyertons, Hood, Kivlin, Kowert & Goetzel P.C., Deposit Account No. 50-1505/5150-83001/JCH.

Also filed herewith are the following items:

☐ Request for Continued Examination

☐ Terminal Disclaimer

☐ Power of Attorney By Assignee and Revocation of Previous Powers

☐ Notice of Change of Address

☐ Other:

Respectfully submitted,

/Jeffrey C. Hood/

Jeffrey C. Hood, Reg. #35198
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel PC
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800
Date: 2008-02-29 JCH/MSW